REMARKS

This Amendment is in response to the Office Action dated February 12, 2003. Applicants submit herewith a petition for a one month extension to respond, extending the response due date from May 12, 2003 to June 12, 2003.

Claims 1-20 are pending in the present application. Claims 1, 11, 12, 13, 14 18, 19 and 20 have been amended, and claim 10 has been canceled. Accordingly, claims 1-9 and 11-20 remain pending in the present application.

Amended Claims

Applicants thank the Examiner for the telephonic interview on May 27, 2003. Applicants have amended independent claims 1, 18, 19 and 20 to clarify the present invention. Specifically, the claims were amended to include the element recited in claim 10, which provides that "at least one of the tree structures represents an output data structure wherein any associated list item defines a formatting definition." Support for this amendment is found in claim 10. No new matter has been presented.

The remaining claims were amended to correct claim dependencies.

35 U.S.C. § 102 Rejections

The Examiner rejected claims 1, 2, 4-7, 8-11, 13-16, and 18-20 under 35 U.S.C. §102(b) as being anticipated by Premerlani et al. (U.S. Patent No. 5,555,367). In so doing, the Examiner stated:

> Referring to claims 1, 18, 19 and 20, Premerlani discloses a tool for graphically defining an expression (column 1, lines 9-11). Premerlani also discloses a graphic user interface which is responsive to user input for defining a tree structure comprising a hierarchical series of nodes (column 4, lines 25-29) and lists comprising a plurality of items with each item being associated with

> respective node of an associated tree (Figure 2). Premerlani discloses that an expression is generated after reading the graphic definition of the expression provided by a user through the GUI component (column 2, lines 35-37). Premerlani also discloses that the graphic definition is analyzed to generate an expression based on the structure of the tree and the list items associated with the respective nodes of a tree (column 2, lines 55-57).
>
> . . . .
>
> Referring to claim 10, Premerlani discloses allowing user to further define a tree structure representing an output data structure wherein any associated list item defines a formatting definition (column 4, lines 50-54 and column 5, lines 5-6).

Applicants respectfully traverse. The present invention is directed to a method and tool for graphically defining an expression, in particular, a query for complex or non-native data types. In the prior art, graphical user interface tools allow a developer to formulate expressions for configuring either filters for incoming messages (in a messaging and queuing system) or database queries. Those tools, however, only allow users to define values for simple data types.

According to the preferred embodiment of the present invention, the tool includes a graphic user interface (GUI) that allows a user to define one or more tree structures comprising a hierarchical series of nodes. At least one of the tree structures represents an output data structure. The user also is allowed to define one or more lists that include a number of items, whereby each item on a list is associated with a node on a tree. Any item associated with the tree structure representing the output data structure defines a formatting definition.

Once the user has defined the one or more tree structures, an expression generator analyzes the tree/list structure and generates an expression. The expression is used to configure modules in a relational message broker or to configure a database query.

The present invention, as recited in claims 1 and 18, provide:

> 1. A tool for graphically defining an expression, said tool comprising:
> a graphic user interface (GUI) component comprising:
> means, responsive to user input, for defining one or more tree structures comprising a hierarchical series of nodes, and one or more lists

comprising a plurality of items, each list item being associated with a respective node of an associated tree structure, wherein at least one of the tree structures represents an output data structure wherein any associated list item defines a formatting definition;

an expression generator component adapted to read a graphic definition of an expression provided by a user through said GUI component, expression generator component comprising:

means for analyzing said graphic definition and generating an expression based on the structure of each tree and any list items associated with respective nodes of a tree.

18. A method for graphically defining an expression in accordance with a graphic definition comprising the steps of:

(a) defining one or more tree structures comprising a hierarchical series of nodes, and one or more lists comprising a plurality of items responsive to user input, each list item being associated with a respective node of an associated tree structure, wherein at least one of the tree structures represents an output data structure wherein any associated list item defines a formatting definition;

(b) analyzing said definition; and

(c) generating an expression based on the structure of each tree and any list items associated with respective nodes of a tree.

Claims 19 and 20 are computer product and system claims, respectively, that are similar in scope to claim 18.

In contrast, Premerlani is directed to using an object-oriented diagram to represent a database query and automatically creating the code for the query from the object-oriented diagram. The Examiner equates the present invention's tree structures with Premerlani's object-oriented diagram illustrated in Figure 2. The Examiner considers each object class 22 to represent a hierarchical node and that the attributes associated with the object class are equivalent to the list items. Applicants respectfully disagree.

Applicants respectfully submit that Premerlani fails to teach or suggest defining at least one tree structure that "represents an output data structure wherein any associated list item defines a formatting definition," as recited in claims 1, 18-20. According to Premerlani, the object-oriented relational diagram in Figure 2 "models a particular data structure *within a database*." (Col. 4, lines 49-50). Premerlani's object-oriented diagram cannot and does not

represent "an output data structure." Furthermore, even if it could, the list of attributes for an object class "are data *values* for the object class." (Col. 3, lines 60-66). As such, they cannot and do not define "a formatting definition," as recited in claim 1, 18-20.

The portions of Premerlani cited by the Examiner supporting this feature disclose that "[t]he object-oriented diagram models a particular data structure within a database. A query is specified for the object diagram at 46. At 48, a series of transformations (i.e., retain, duplicate, merge, constrain, restrict, and bind) are performed on the object model by using the mouse. The transformations are additional operations that can be added to OMTool" (col. 4, lines 50-54), and "[t]he retain operation retains only the selected classes and associations" (col. 5, lines 5-6). The cited portions discuss how the object-oriented relational diagram is transformed into the revised diagram. The revised diagram is used to generate the computer code for the query.

Applicants respectfully submit that Premerlani's revised diagram (see e.g., Figure 7b and Figure 8c) does not "represent an *output* data structure" because it represents *the query*. Moreover, the attributes for the class objects that make up the revised diagram "are data *values* for the object class" (col. 3, lines 60-66), not formatting definitions. Thus, Applicants respectfully submit that there is no teaching or suggestion of defining at least one tree structure that "represents an output data structure wherein any associated list item defines a formatting definition," as recited in claims 1, 18-20.

Based on the arguments above, Applicants respectfully submit that Premerlani fails to teach or suggest the cooperation of elements recited in claims 1, 18, 19 and 20. Thus, claims 1, 18, 19 and 20 are allowable over the cited reference. Claims 2, 4-7, 8, 9, 11, and 13-16 depend on claim 1, and the above arguments apply with full force. Therefore, claims 2, 4-7, 8, 9, 11, and 13-16 are also allowable over the cited reference.

35 U.S.C. §103 Rejections

The Examiner rejected claim 3 under 35 U.S.C. §103(a) as being unpatentable over Premerlani and Gawlick et al (U.S. Patent No. 6,377,953). Claim 12 was rejected as being unpatentable over Premerlani. Claim 17 was rejected as being unpatentable over Premerlani and Moshfeghi (U.S. Patent No. 6,476,833). Claims 3, 12 and 17 depend on claim 1 and therefore, are allowable because they depend on an allowable base claim. Accordingly, Applicants respectfully submit that claims 3, 12 and 17 are also allowable over the cited references.

Conclusion

In view of the foregoing, it is submitted that the claims 1-9 and 11-20 as now presented, are allowable over the cited references and are in condition for allowance. Applicants respectfully request reconsideration of the rejections and objections to the claims, as now presented.

Applicants believe that this application is in condition for allowance. Should any unresolved issues remain, Examiner is invited to call Applicants' attorney at the telephone number indicated below.

Respectfully submitted,

June 2, 2003
Date

Joyce Tom
Sawyer Law Group LLP
Attorneys for Applicant(s)
Reg. No. 48,681
(650) 493-4540